

Frickeln

Dominik Bacher, Max Mössinger

Computer Engineering
HS-Furtwangen

11. November 2010

Gliederung

- 1 **Binärtools**
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Gliederung

- 1 Binärtools
- 2 sourcebrowser
- 3 VIM Kleinigkeiten
- 4 Makefiles und Autotools
- 5 GCC
- 6 codeanalyse
- 7 Debugging
- 8 Sonstiges

Binary Tools

strings

- findet ASCII-Zeichenketten
- z.B. finden von Fehlermeldungen
- findet in /dev/mem manchmal Passwörter

hexdump

gibt eine Datei im Hex-Format aus

bvi

binary VI

Binary Tools

strings

- findet ASCII-Zeichenketten
- z.B. finden von Fehlermeldungen
- findet in /dev/mem manchmal Passwörter

hexdump

gibt eine Datei im Hex-Format aus

bvi

binary VI

nm objdump

- Analyse von Binarydateien

objdump

- h gibt Section header aus
- t gibt printet Symbole aus
- d disassembler

Sourcebrowser

ctags

- Erstellt ein Index eines Verzeichnisses
- suche nach Definitionen und Deklarationen
- vim plugin

ctags vim

- `ctags -R *`
- `:tags` zeigt Tag-Stack an
- `:help tags-and-searches`
- `ctrl]` - jump from call to definition
- `ctr t` - jump back

Sourcebrowser

ctags

- Erstellt ein Index eines Verzeichnisses
- suche nach Definitionen und Deklarationen
- vim plugin

ctags vim

- `ctags -R *`
- `:tags` zeigt Tag-Stack an
- `:help tags-and-searches`
- `ctrl]` - jump from call to definition
- `ctr t` - jump back

cscope

- text/ncurses basierend
- C++ und Java
- VIM Support :-)

TagList

- Source Code Browser
- <http://vim-taglist.sourceforge.net>

vimdiff

- Teil von VIM
- Stellt diff in mehreren Fenster nebeneinander dar
- Farbig :-)

Makefiles

- Erkennt selbständig Änderungen
- Selbst Definierbare abhängigkeiten von Projektzweigen
- umsetzung von Pre und Post Build Events

Tragets

`make all` Buildet alles (Programme, libraries, doc, ...) (Same as make)

`make install` Installiert das Paket

`make install-strip` Gleich wie `make install` entfernt dann debugging symbole

`make uninstall` Gegenteil von `install`

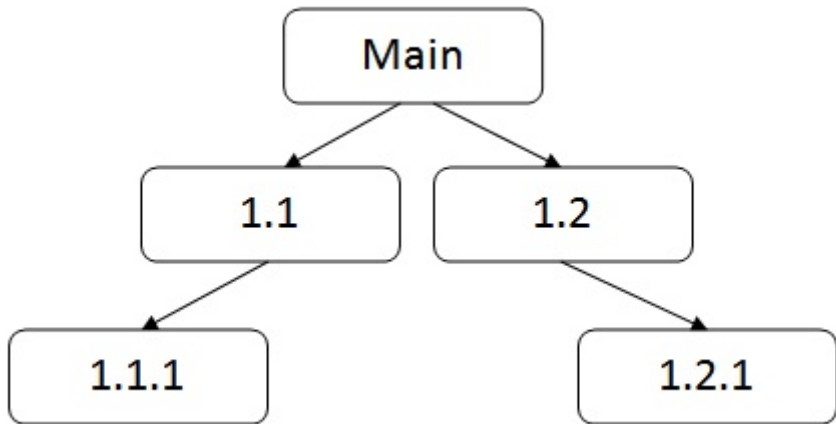
`make clean` Aufräumen

`make distclean` Vollständiges aufräumen

`make check` Tests laufen lassen

`make installcheck` Tests zur installation laufen lassen

`make dist` Erstellen eines `PACKAGE-VERSION.tar.gz`.



Autotools

Autoconf Generiert das 'configure' shell script, welches das System zur compiletime analysiert zb. ob gcc oder cc verwendet werden soll.

Automake Generiert Makefiles aus den gesammelten Informationen der configure.ac (Autoconf)

Libtool Generiert Plattform unabhängige Libraries

Compiling

- ./configure
- make
- make install

Warnings

- w unterdrückt Warnings
- Wall zeigt alle Wichtigen Warnings
- Wextra zeigt noch mehr Warnings
- std=... definiert den zu verwendenden C Standard
- pedantic zwingt die Einhaltung des Standards
- pedantic-errors behandelt die Warnings als Fehler
- Werror behandelt ALLE Warnings als Fehler

GCC `__attribute__()`

- ermöglicht besondere Eigenschaften für Variablen und Funktionen zu deklarieren
- mehrere durch Komma getrennte Attribute

bei Funktionen nur bei der Funktionsdeklaration möglich

```
int foo(void) __attribute__((noreturn));
```

bei Variablen vor der Initialwertzuweisung

```
int foo __attribute__((unused)) = 23;
```

lint

- statische Code-Analyse
- splint Linux-Implementierung (kann mehr)

valgrind

- Laufzeitanalyse und Debugging von Programmen
- verschiedene Analyse-Funktionalitäten speziell zum auffinden von Memory-Leaks

gprof

- Zeigt eine Programmstatistik
- Statistik ber Funktionsaufrufe
- Statistik ber Zeitkosten
- gcc muss dafür mit `gcc -pg` compiliert werden
- beim ausführen des Programms wird `gmon.out` erzeugt die gprof analysiert

strace

- Monitoring von System Calls und Signalen
- ermöglicht:
 - Monitoring von System Calls inklusive deren Parametern und Rückgabewerte
 - Anzeigen von empfangenen Signalen
 - Einhängen in einen bereits laufenden Prozess
 - Filter auf bestimmte Events
- Ausgabe in Datei möglich - VIM kennt filetype=strace

ltrace

- ähnlich wie strace
- Monitoring von Library Calls
- funktioniert aber nur mit dynamisch gelinkten Programmen

Debugger

GDB

- gcc -g erzeugt Debugging Symbole
- remote Debugging mit gdbserver (seriell oder TCP)
- ddd DataDisplayDebugger

ddd

- DataDisplayDebugger
- braucht einen untergeordneten Debugger
 - GNU Debugger
 - Perl (perl -d)
 - Bash (bashdb)
 - Python (pydb)
 - GNU Make Debugger (remake)
 - dbx
 - ladebug

lxr

- HTML Source Code Cross Reference

doxygen

- generiert Doku :-)
- unterstützt C, C++, Java, Ada, VHDL...
- HTML, LATEX, Man Pages... als output

javadoc

gleich wie doxygen allerdings nur für java

Fragen?

thx to ciro and n0-1 frickeln 2008